October 2003 UILU-ENG-03-2222
DC-210

Distributed Algorithms for Voronoi Diagrams and Applications in Ad-hoc Networks

Christoforos N. Hadjicostis and Min Cao

Coordinated Science Laboratory
1308 West Main Street, Urbana, IL 61801
University of Illinois at Urbana-Champaign

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. **REPORT DATE**	3. REPORT TYPE AND DATES COVERED
	October 30, 2003	

4. TITLE AND SUBTITLE
Distributed Algorithms for Voronoi Diagrams and Applications in Ad-hoc Networks

5. FUNDING NUMBERS

6. **AUTHOR(S)**
Christoforos N. Hadjicostis and Min Cao

7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
1308 West Main Street
Urbana, Illinois 61801-2307

8. **PERFORMING RGANIZATION REPORT NUMBER**
UILU-ENG-03-2222
(DC- 210)

9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
National Science Foundation
Washington, DC 20050

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. **SUPPLEMENTARY NOTES**
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official position, policy or decision, unless so designated by other documentation

12a. **DISTRIBUTION/AVAILABILITY STATEMENT**
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Voronoi diagram is a fundamental structure in computational geometry and arises naturally in many applications including wireless networking. In this paper, we propose a distributed algorithm by which each node *u* can compute its Voronoi region in *O(d(u))* time, where *d(u)* is the number of the Voronoi neighbors of node *u*. Then we show how the algorithm can be applied in topology control of wireless ad-hoc networks, and also propose a revised version of the algorithm to minimize transmission energy consumption. Further applications of the algorithm in different areas are expected.

14. **SUBJECT TERMS**
Voronoi diagram, distributed algorithms, Ad-hoc networks, topology control

15. **NUMBER OF PAGES**
12 pages

16. PRICE CODE

17. **SECURITY CLASSIFICATION OF REPORT**	18. **SECURITY CLASSIFICATION OF THIS PAGE**	19. **SECURITY CLASSIFICATION OF ABSTRACT**	20. **LIMITATION OF ABSTRACT**
UNCLASSIFIED	**UNCLASSIFIED**	**UNCLASSIFIED**	**UL**

Distributed Algorithms for Voronoi Diagrams and Applications in Ad-hoc Networks

Min Cao and Christoforos Hadjicostis

Abstract

The Voronoi diagram is a fundamental structure in computational geometry and arises naturally in many applications including wireless networking. In this paper, we propose a distributed algorithm by which each node u can compute its Voronoi region in $O(d(u))$ time, where $d(u)$ is the number of the Voronoi neighbors of node u. Then we show how the algorithm can be applied in topology control of wireless ad-hoc networks, and also propose a revised version of the algorithm to minimize transmission energy consumption. Further applications of the algorithm in different areas are expected.

1 Introduction

The Voronoi diagram has been a central subject in computational geometry with many applications in various areas, such as biology, ecology, geography, physics, archaeology, crystallography [1]. Recently, the Voronoi diagram has become a useful tool for exploring location and routing issues in wireless networks and has given rise to the problem of distributed computation of Voronoi diagrams [2].

In the two-dimensional case, the Voronoi diagram is a tesselation of the plane. Given a set of n nodes $\mathcal{V} = \{v_1, \ldots, v_n\}$ in the plane, we associate each point in the plane with the closest node in the set $\mathcal{V}$. The result is a tessellation of the plane into a set of polygons, $V_1, \ldots, V_n$, associated with nodes $v_1, \ldots, v_n$, respectively. We call this tessellation the *Voronoi diagram* and the polygons $V_1, \ldots, V_n$ constitute the *Voronoi polygons* or the *Voronoi regions*. To state the definition mathematically, let $\mathbf{x}_i = (x_i, y_i)$ be the location vector of node v_i on the two-dimensional Euclidean space. The Voronoi polygon associated with v_i is defined as

$$V_i = \bigcap_{j:j\neq i} \{\mathbf{x} \in \mathbb{R}^2 \mid \|\mathbf{x} - \mathbf{x}_i\| < \|\mathbf{x} - \mathbf{x}_j\|\}.$$

In many situations, people also consider the Voronoi diagram of a set of nodes within a bounded region S (for example a square region in the plane). In such case, the Voronoi polygon associated with v_i is

$$V_i = \bigcap_{j:j\neq i} \{\mathbf{x} \in S \mid \|\mathbf{x} - \mathbf{x}_i\| < \|\mathbf{x} - \mathbf{x}_j\|\}.$$

A node that defines a segment of the Voronoi polygon of node u is called the *Voronoi neighbor* of u. We denote the set of Voronoi neighbors of node u as $N(u)$.

To construct the Voronoi diagram efficiently, many algorithms have been developed in computational geometry [3, 4, 5]. The two major methods are the divide-and-conquer method and the incremental method [6]. The computational time of the former method is $O(n \log n)$ in the worst case and on the average. The later method, more specifically, the quaternary incremental algorithm in [4] runs in time $O(n^2)$ in the worst case, and $O(n)$ on the average. All of the above mentioned algorithms are centralized in nature, however, despite the fact that many emerging applications require a distributed algorithm. One such example is wireless ad-hoc networks where no central unit exists to perform the global computational task; in such a network, each node has to compute its own Voronoi region locally. More importantly, there is a large communication cost associated with getting information to and from nodes. Although centralized algorithms for the computation of Voronoi diagrams have been extensively studied, distributed algorithms have not been formerly stated. In this paper, we propose an algorithm that computes the Voronoi regions of each node in a distributed manner.

The rest of the paper is organized as follows. In Section 2, we present our distributed algorithm for the computation of Voronoi regions. Applications in topology control of ad-hoc networks are discussed in Section 3. In Section 4, we consider the energy consumption involved in communication among the nodes and propose a revised algorithm for ad-hoc networks to gain energy efficiency. We conclude in Section 5 with a summary and directions for future research.

2 Distributed Algorithm

Let $\mathcal{V}(u) = \mathcal{V} - \{u\}$. For each node v in $\mathcal{V}(u)$, let v' be the midpoint of the segment uv that connect nodes u and v, as illustrated in Figure 1. Let $l_{v'}$ be the line that passes through point v' and is perpendicular to the segment uv, and l_v be the line that passes through point v and is perpendicular to the segment uv. We denote the half plane that is defined by line $l_{v'}$ and contains node u as $H_{v'}$, and the half plane that is defined by line l_v and contains node u as H_v. Then the *Voronoi region* of node u is

$$V(u) = \bigcap_{v \in \mathcal{V}(u)} H_{v'},$$

which is a polygon in the two-dimensional space. We also define the *Enclosure region* in a similar fashion as

$$E(u) = \bigcap_{v \in \mathcal{V}(u)} H_v.$$

The Voronoi region and the Enclosure region of a point u are illustrated in Figure 1. They are similar in shape, with the vertices of the Voronoi region located at half way from the vertices of the Enclosure region towards node u. Clearly it is equivalent to find the Voronoi region or the Enclosure region. For convenience, in the following we will discuss the computation of the Enclosure region instead of the Voronoi region.

We consider the case where there are n nodes in a bounded region S. For notational simplicity, let $v_1, v_2, \ldots, v_{n-1}$ be the nodes in $\mathcal{V}(u)$ ordered by ascending distance away from node u. The way our algorithm work with is by having each node look at its neighborhood (i.e., nodes within a certain radius) and keep enlarging this neighborhood until it has full knowledge of its own Voronoi region.

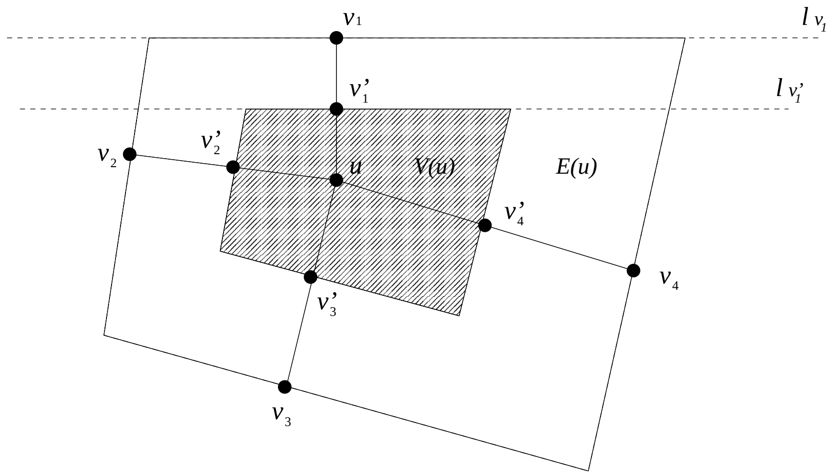

Figure 1: The Voronoi region and Enclosure region of node u.

A centralized procedure for the calculation of the Enclosure region $E(u)$ can be described as follows:

(i) first, calculate the intersection region of S and the half plane defined by the nearest node v_1, that is $E_1(u) = S \cap H_{v_1}$;

(ii) then, consider the second nearest node v_2 is taken into account and we set $E_2(u) = E_1(u) \cap H_{v_2}$;

(iii) continue the above procedure until we get $E_{n-1}(u)$.

Clearly, $E(u) = E_{n-1}(u)$ and each of the steps in this procedure involves only basic geometric computations. The procedure requires n steps to find the Enclosure region of node u, and clearly the order with which neighboring nodes are considered does not matter.

In most cases only a limited number of neighboring nodes affects the Enclosure region. The natural question is then to determine these nodes in an efficient manner. Here, we find a simple criterion that provides the answer to this question. Let r_i be the maximum distance to node u in $E_i(u)$

$$r_i = \max_{x \in E_i(u)} \|\mathbf{u} - \mathbf{x}\|$$

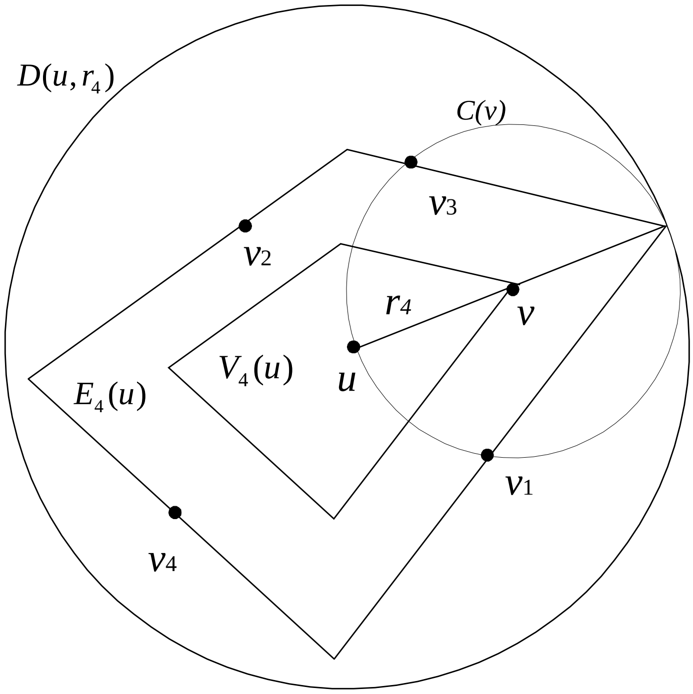

Figure 2: Stopping criterion of the computation of the Enclosure region.

Let $D(u, r_i)$ be the disk centered at u with radius r_i. And let $V_i(u)$ be the corresponding Voronoi region defined by $v_1, \ldots, v_i$

$$V_i(u) = \bigcap_{j=1,\ldots,i} H_{v'_j}$$

Then the following theorem holds.

Theorem 1 *If there are no other nodes inside disk $D(u, r_i)$ except $v_1, \ldots, v_i$ and u, then $E_i(u)$ $(V_i(u))$ is the Enclosure region $E(u)$ (Voronoi region $V(u)$) of node u.*

Proof. Consider a vertex of $V_i(u)$, denoted by v, as illustrated in Figure 2. Let $C(v)$ be the disk that centered at v with radius $r_i/2$. By the property of Voronoi cells [7], $C(v)$ contained two nodes, say v_j, v_k on its boundary, and none of the nodes $v_1, \ldots, v_i$ can be in the interior of $C(v)$. Since there are no other nodes inside disk $D(u, r_i)$ except $v_1, \ldots, v_i$ and u, and note that $C(v) \supset D(u, r_i)$, the interior of $C(v)$ must be empty. Then by the theorem in [7], v is a vertex of $V(u)$. This is true for any vertex v of $V_i(u)$, so $V_i(u) = V(u)$. Correspondingly, $E_i(u) = E(u)$. $\square$

The above theorem provides a criterion for stopping the exploration of new nodes in the centralized algorithm described in the beginning of this section. We can now state our distributed algorithm as follows.

Algorithm 1 *Calculation of Enclosure region(u)*

Set $E_0(u) = S,\ i = 1$

WHILE TRUE

1. $E_i(u) = E_{i-1}(u) \cap H_{v_i}$
2. $r_i = \max_{x \in E_i(u)} \|\mathbf{x} - \mathbf{u}\|$
3. let $D(u, r_i)$ be the disk centered at u with radius r_i
4. if $\{v|\ v\ \text{inside} D(u, r_i)\} - \{u\} \cup \{v_1,\ \ldots,\ v_i\} = \emptyset$, then BREAK, else $i = i + 1$

END

$E(u) = E_i(u)$, and $D(u) = D(u, r_i)$

From the definition of the Enclosure region, we can see that $E_1(u) \supseteq E_2(u) \supseteq \cdots \supseteq E(u)$. The above algorithm stops when $D(u, r_i)$ is just large enough to enclose $E(u)$. It is possible that $D(u, r_i)$ contains additional nodes (other than $v_1,\ \ldots,\ v_i$) when $E_i(u) = E(u)$, as illustrated in Figure 3. These nodes, however, do not affect $E(u)$, otherwise $E(u)$ cannot be the Enclosure region. In such cases, the algorithm needs to go a few more steps to finish. If all nodes are randomly distributed, the number of such nodes can be neglected, and the time complexity of the above algorithm for each node u is $O(d(u))$, where $d(u)$ is the number of Voronoi neighbors of node u. This should be contrasted with the basic centralized procedure whose time complexity is $O(n)$, where n is the total number of nodes in S. When the nodes

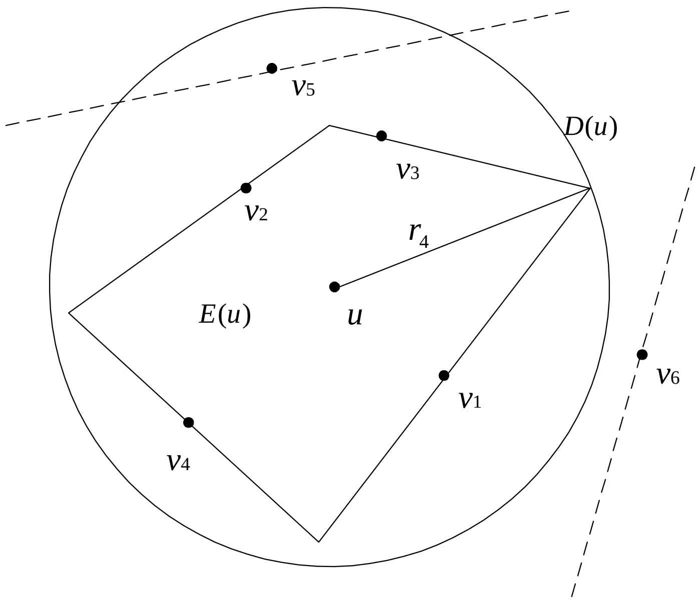

Figure 3: Computing the Enclosure region of node u.

are well-spaced, the number of voronoi neighbors of a node u at typically 6, so $d(u)$ is expected to be a constant, and independent of the total number of nodes. In a wireless network, where there are hundreds of nodes, it is clearly more efficient to employ the distributed algorithm at each node rather than the centralized procedure.

3 Applications in Ad-hoc Networks

A mobile *ad-hoc* network is an autonomous system of mobile nodes connected by wireless links [8]. In such a network, there is no static infrastructure such as base stations; instead, hosts that are not within radio range need to communicate through one or more intermediate nodes that act as routers. The nodes in ad-hoc networks are free to move around randomly, which requires routing protocols to be adaptive and able to maintain routes in spite of the changing network connectivity [9].

A wireless network can be modeled as a directed graph $G = (\mathcal{V}, E)$, where $\mathcal{V}$ is the set of mobile nodes, and edge $(u, v) \in E$ if node v is in the transmission range of node u. We call G the *transmission graph* and assume that it is strongly connected. Then the routing protocol runs a distributed shortest-path algorithm on G_t to find shortest path between each source-destination node pair [10].

Instead of using the transmission graph G, Li shows that the enclosure graph G_e also contains all the shortest paths and can guarantee connectivity, while at the same time it is sparse and greatly reduces the time and space complexity of shortest-path algorithms. The enclosure graph G_e is formed by connecting each node to its Voronoi neighbors. In order to implement the approach in [10] in a distributed manner, we need distributed versions of the algorithms for the calculation of the shortest paths and the Voronoi regions. While computing the Voronoi regions, the algorithm described in Section 2 also finds the neighboring nodes that affect the Voronoi region of each node u. Thus the algorithm in Section 2 is applicable in this context; what is more important, this algorithm can also adapt to the mobility of the nodes.

In mobile wireless networks, the networking protocol must be able to dynamically update its links in order to maintain connectivity. Notice that a node that moves from one position to another can be viewed as two events: one node is deactivated at the old position and one node is activated at the new position [10]. Therefore, in the following, we only consider how to add a new node to the network and how to remove one node from the network.

First we assume that a node w is added to the network. It is easy to show that only the Voronoi regions of nodes u whose $D(u)$ contain w can be affected. To update the networking topology, node w broadcasts its position information to nearby nodes. Each node u that receives the message checks whether w is in $D(u)$. If $w \notin D(u)$, $E(u)$ will not be affected. If $w \in D(u)$, set $E(u) = E(u) \cap H_w$. And if $E(u) \cap \widetilde{H}_w \neq \emptyset$, w is added to $N(u)$, and those nodes that no longer define a border of $E(u)$ are deleted from $N(u)$. This update needs only one step of computation.

Then we consider how to remove a node from the network. Removing node w will only affect a node u if $w \in N(u)$. And it may introduce some new neighbors to node u. Suppose w is formerly the i_{th} nearest nodes from u, and $E_i(u)$ is the Enclosure region defined by $v_1, \ldots, v_{i-1}$, then we set $i = i + 1$, and resume Algorithm 1 at Step 2 until the algorithm stops. Thus $E(u)$ and $N(u)$ are updated. In the worst case, the update can be done in $O(d(u))$ time.

From the above discussion we see that our algorithm is adaptable to dynamically changing networks. One of the main advantages of our algorithm is that it can use as much previous information as possible and avoid unnecessary re-computation.

4 Considering Communication Energy Consumption

The algorithm described in Section 2 is mostly concerned with reducing the time complexity of distributed computations of Voronoi regions. The algorithm incrementally enlarges the neighborhood of node u by adding a next nearest node and by updating the Enclosure region each time, until the neighborhood is just large enough to determine the Voronoi region. The algorithm does not consider any nodes that are not used in determining the Voronoi region. Therefore, it is efficient in the sense of reducing time complexity.

The algorithm described in Section 2, however, require full knowledge of the positions of all nodes in its neighborhood in advance in order to perform the computation incrementally. Therefore, when this algorithm is applied to ad-hoc networks, each node u will need to first get position information from all the nodes in its transmission range. The power consumption incurred is proportional to R^α, where R is the maximum transmission range of node u, and α is the path loss factor. This energy consumption can be very large. Since wireless nodes are generally powered by batteries, energy efficiency is a critical issue in ad-hoc networks. Hence, in this section we propose a revised version of the algorithm developed in Section 2 that takes transmission energy consumption into consideration.

Let $N_r(u)$ be the set of neighboring nodes of u that is inside disk $D(u, r)$, where r is a parameter we can choose. Denote the set of neighboring nodes in the transmission range of u as $R(u)$, and then

$$N_r(u) = \{v \mid v \in R(u), v \text{ inside } D(u, r)\}.$$

If $E(u, N_r(u))$ denotes the Enclosure region of u defined by the neighboring nodes in $N_r(u)$

and d denotes the maximum distance to node u in $E(u, N_r(u))$, i.e.,

$$d = \max_{x \in E(u, N_r(u))} \|\mathbf{u} - \mathbf{x}\|,$$

We have the following theorem which is a variation of Theorem 1.

Theorem 2 *$E(u, N_d(u))$ is the Enclosure region $E(u)$ of node u.*

Proof. We label all the neighboring nodes of u in $N_d(u)$ as $v_1, \ldots, v_i$ by ascending distance from u. Hence $E(u, N_d(u)) = E_i(u)$. It is obvious that there are no other nodes inside disk $D(u, d)$ except $v_1, \ldots, v_i$ and u. So by Theorem 1, $E(u, N_d(u)) = E_i(u) = E(u)$. $\square$

Hence, we can calculate the Enclosure region of node u by first computing $E(u, N_r(u))$ and then updating the Enclosure region with the additional nodes in $N_d(u) - N_r(u)$. At the beginning of each step, we get position information from nodes in $N_r(u)$ and $N_d(u) - N_r(u)$ respectively. Therefore, in both steps, we can compute or update the Enclosure region using the incremental algorithm of Section 2.

The problem with the above approach is that if r is not chosen properly, d might be very large and may lead to excessive power consumption. To avoid this, we can use the following strategy: if $d \leq cr$, then increase the radius of $D(u)$ to d, else if $d > cr$, then increase the radius to cr, where c is another parameter we can set. Now we describe the algorithm as follows.

Algorithm 1 *Calculation of Enclosure region(u)-revised*

1. Set r, c
2. $N_r(u) = \{v \mid v \in R(u), v \text{ inside } D(u, r)\}$, $E_r(u) = \bigcap_{v \in N_r(u)} H_v$
3. If $E_r(u)$ is inside $D(u, r)$, then $E(u) = E_r(u)$, BREAK
4. $d = \max_{x \in E(u, N_r(u))} \|\mathbf{u} - \mathbf{x}\|$
5. If $d \leq cr$, then $E(u) = E_d(u)$, BREAK

6. Else $r = cr$, goto 2.

The transmission power consumption with the above algorithm is $r^\alpha + (cr)^\alpha + (c^2 r)^\alpha + \cdots$. We need to choose c, r so that $\mathbf{E}\{r^\alpha + (cr)^\alpha + (c^2 r)^\alpha + \cdots\}$ is minimized. We will leave this problem to future study.

5 Conclusion

We have described a distributed algorithm for the computation of a Voronoi diagram. By employing a stoping criterion, the algorithm is able to find the Voronoi region of each node in $O(d(u))$ time, where $d(u)$ is the number of Voronoi neighbors of node u. We have also shown how the algorithm can be applied to topology control of wireless ad-hoc networks, especially in the case when the network is dynamically changing. By considering the transmission energy consumption in ad-hoc networks, we further proposed a revised version of the algorithm to minimize energy consumption when computing the Voronoi diagram.

There are many other problems in various application areas, where the distributed computation of Voronoi diagrams is needed. For example, Bullo in [12] incorporates the above algorithm in solving the coverage control problem for mobile sensing networks. We are expecting further study and application of the distributed computation of Voronoi regions.

References

[1] A. Okabe, B. Boots and K. Sugihara, *Spatial Tesselations: Concepts and Applications of Voronoi Diagrams*, John Wiley & Sons, Chichester, 1992.

[2] I. Stojmenovic, “Voronoi diagram and convex hull based geocasting and routing in wireless networks,” Technical report, University of Ottawa, TR-99-11, December 1999.

[3] S. Fortune, “A sweepline algorithm for Voronoi diagrams,” *Algorithmica*, 2, 153–174, 1987.

[4] T. Ohya, M. Iri and K. Murota, "Improvements of the incremental method for the Voronoi diagram with computational comparisons of various algorithms," *Journal of the Operations Research Society of Japan*, 27, 306-336, 1984.

[5] S. Fortune, "Voronoi diagrams and Delaunay triangulations," *Computing in Euclidean geometry*, D. Z. Du and F. Hwang eds, 225-265, World Scientific, Singapore, 1995.

[6] F. P. Preparata and M. I. Shamos *Computational Geometry - an Introduction*, Springer Verlag, 1985.

[7] M. Berg, M. Kreveld, M. Overmars and O. Schwarzkopf *Computational Geometry - Algorithms and Applications*, 148-151, Second edition, Springer Verlag, 1998.

[8] P. Macker and M. S. Corson, "Mobile ad hoc networking and the IETF," *ACM Mobile Computing and Communications Review*, vol. 2, no. 2, 1998.

[9] E. M. Royer and C.-K. Toh, "A review of current routing protocols for ad-hoc mobile wireless networks," *IEEE Personal Communications Magazine*, 46–55, April 1999.

[10] L. Li and J. Hapern, "Minimum energy mobile wireless networks revisited," In *IEEE International Conference on Communications(ICC)*, June 2001.

[11] X.-Y. Li, P.-J. Wan, Y. Wang, and O. Frieder, "Sparse power efficient topology for wireless networks," In *Proc. IEEE Hawaii Int'l Conf. System Sciences (HICSS)*, 2002.

[12] J. Cortes, S. Martinez, T. Karatas, F. Bullo, "Coverage control for mobile sensing networks: Varations on a theme," In *Mediterranean Conference on Control and Automation*, Lisbon, Portugal, 2002.